Преимущества грамматики связей для русского языка

С. В. Протасов

Московский Физико-Технический Институт[1]

Аннотация

Автором была разработана формальная грамматика, названная грамматикой связей для русского языка. Грамматика связей не является ни грамматикой составляющих, ни грамматикой зависимостей. В докладе показано, каким образом русскоязычные предложения могут кодироваться и анализироваться с помощью данного контекстно-свободного формализма и объясняются основные преимущества данного подхода: это более мощное описание языка, существование открытого и эффективного базового алгоритма разбора.

1 Введение

Задача семантической обработки информации текстов является одной из основных в компьютерной лингвистике. Как правило, в качестве главного средства для семантического анализа выступают специализированные компьютерные анализаторы (программы-анализаторы, разбирающие предложения), которые выявляют отношения между словами и их роль в предложении. Разные языки по разному поддаются алгоритмическому разбору предложений. Если в английском языке, используя грамматические правила, выявить связи и отношения между словами довольно легко, то в русском языке задача значительно усложняется, так как свободный порядок слов создает экспоненциальный взрыв перебора вариантов для большинства алгоритмов разбора. Мы можем легко построить анализатор (например, на LISP или Scheme), который знает много грамматических явлений, но с русским языком он будет работать очень долго, начиная с предложений из 7-10 слов. Либо мы можем ограничиться небольшим набором грамматических явлений, анализатор будет работать быстро, но на реальных текстах процент разбора будет небольшой. Данную проблему в большинстве алгоритмов решают за счет использования семантических словарей, причём составляемых вручную. Данные словари используют в основном в целях сужения числа разрешенных правил *“по умолчанию”*, для слов, отсутствующих в семантических словарях. Искусственное ограничение на связи приводит к значительному ухудшению качества разбора и их сильной зависимости от качества и полноты словарей. Однако возможен и другой путь - попытаться использовать более эффективные алгоритмы (возможно, они будут более требовательны к объему памяти) и описывать более широкую грамматику. Более широкая грамматика позволит позже составлять семантические словари через специальные статистические алгоритмы. Узкая грамматика и анализатор никак не позволяют создавать семантические словари, требуемые для расширения грамматики. Если у нас в грамматике нет творительного падежа - то составить словарь глаголов с “творительной” валентностью мы не можем. Имея заданную грамматику и анализатор, мы имеем возможность разбивать слова на семантические классы только в пределах грамматических явлений уже описанных данной грамматикой. Хотелось бы найти такую грамматику и такой алгоритм, что бы выполнялись условия как достаточной широты грамматики, так и условие приемлемой скорости работы алгоритма разбора.

Автору удалось найти и адаптировать эффективный алгоритм, обеспечивающий как возможность более свободного порядка слов в русском языке, так и приемлемую скорость разбора предложения. В основе алгоритма лежит лингвистический формализм называемый грамматикой связей.

[1]Россия, Москва 113303, ул. Керченская, д. 1«А», корп. 1, МФТИ

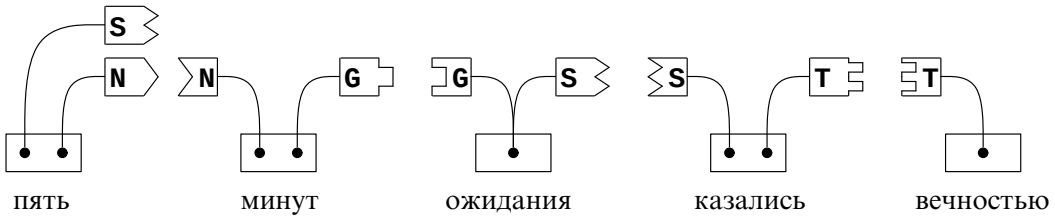
2 Грамматика связей

Большинство предложений в большинстве индоевропейских языков обладают особым свойством, а именно: если провести линии между словами, которые между собой связаны, то эти линии не пересекаются. Это свойство называется проективностью и лежит в основе лингвистического формализма грамматики связей.

Грамматика связей состоит из *слов* (терминальных символов грамматики), которые имеют ограничения или *требования по связям*. Последовательность слов является предложением языка, только в том случае, если все слова соединены, связи не пересекаются и все требования по связям выполнены, то есть выполняются три условия:

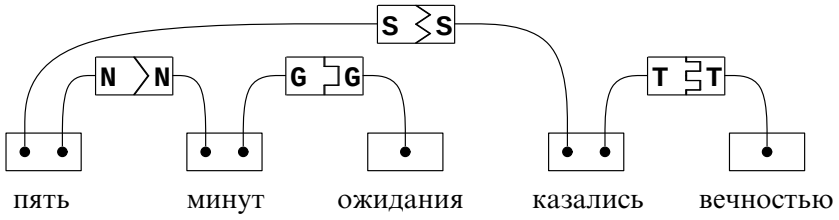
1. Проективность: связи между словами не пересекаются (связи рисуются над словами).
2. Связность: отсутствуют изолированные слова или несвязанные группы слов.
3. Требования: выполнены все условия на связи для каждого слова.

Все требования по связям для каждого слова содержатся в *словарях*. На следующем рисунке показано, как выглядит небольшой словарь из слов *пять,минут,ожидания,казались,вечностью*. Требования по связям для слов изображены в виде схем над ними.



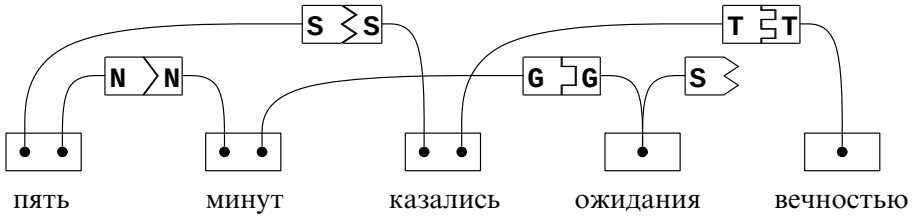
Непрямоугольные блоки в виде разъемов называются *коннекторами*. Коннектор может соединяться только с соответствующим ему коннектором с такой же формой разъема. Причем коннекторы направленные направо могут соединяться только с коннекторами направленными налево и наоборот. Если коннектор соединен, то все другие коннекторы, исходящие из той же черной точки (если они есть) не могут использоваться. Например слово *ожидания* требует либо коннектор G с левой стороны, либо коннектор S с правой стороны, а одновременно их использовать нельзя. Соединенные коннекторы образуют *связь* между словами.

На следующем рисунке показано, как в предложении *Пять минут ожидания казались вечностью* могут быть удовлетворены все требования по связям.



(Неиспользующиеся коннекторы на рисунке не показаны) Таким же образом можно показать, что *ожидания казались вечностью* также является предложением данной грамматики, хотя и не имеет смысла. А последовательность слов *пять ожидания казались вечностью* невозможно соединить между собой правильно, так как отсутствует слово, которое могло бы соединиться со словом *пять* через его обязательный коннектор N. Таким образом слово *пять* изолировано и эта последовательность слов не принадлежит данной модели языка. Аналогичные проблемы у последовательностей *минут казались вечностью* и у *пять минут ожидания*

казались. Последовательность слов *пять минут казались ожидания вечностью* может быть соединена только с пересечением связей.



И поэтому, несмотря на то, что все требования удовлетворены, эта последовательность слов также не является предложением языка грамматики.

Набор связей, который показывает, что последовательность слов принадлежит языку грамматики связей, называется *связкой*. С этого места рисунки схем *связок* будут более простые. Для примера приведем упрощенный рисунок связки, который показывает, что предложение *пять минут ожидания казались вечностью* является частью языка грамматики.

Все требования по связям для каждого слова из языка грамматики могут быть записаны в сжатой математической форме, которая понятна компьютеру. К примеру, следующий словарь кодирует требования по связям для вышеприведенного примера.

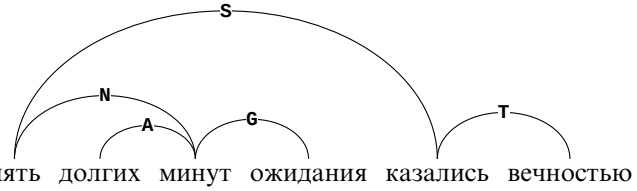
слово	формула
пять	N+ И S+
минут	N- И G+
ожидания	G- ИЛИ S+
казались	S- И T+
вечностью	T-

Требования по связям для каждого слова выражаются логическими формулами с операциями И, ИЛИ над коннекторами или над более простыми формулами. Суффиксы + и − обозначают направление коннекторов (по отношению к слову для которого они определяются), и в этом направлении должен найтись встречный коннектор с противоположным знаком. Если несколько коннекторов или несколько формул объединены оператором И, то требования по связям для полного выражения считаются удовлетворенными только в том случае, если выполнены требования для каждого из слагаемых. Оператор ИЛИ требует выполнения одного и только одного слагаемого. Крайне важен порядок слагаемых в операторе И. Самый левый коннектор в выражении должен быть соединен с самым близким словом. Например для объекта *пять* слово, соединяемое через коннектор $N+$, должно быть ближе, чем слово соединяемое через коннектор $S+$. Таким образом последовательность *пять казались минут* становится запрещенной.

На следующей таблице показан словарь для более сложной грамматики связей. Запись *{вып.}* означает, что выражение не обязательно, и может быть опущено, а запись $@A-$ означает, что в этом месте могут быть соединены один или более коннекторов типа A.

слово	формула
пять	N+ И S+
минут	{@A-} И N- И {G+}
ожидания	G- ИЛИ S+
казались	S- И T+
вечностью	T-
долгих тяжких	A+

В этой грамматике предложение *пять долгих минут ожидания казались вечностью* принадлежит языку, потому как существует следующая связка:



В данном предложение существует только одна связка, удовлетворяющая всем условиям, однако вообще говоря таких связок может быть несколько.

Так как слова языка можно разбивать на классы слов разными способами, и даже для одинаковых классов слов можно определять разные коннекторы и строить связи разных типов, то вообще говоря грамматик связей может быть очень много. Однако для каждой грамматики принадлежащей к классу грамматики связей существует алгоритм и программа разбирающая предложения языка и строящая все разрешенные связки.

Все, что было сказано до сих пор про грамматики связей, было придумано не автором, а американскими учеными (Temperley D. Sleator D. Lafferty J. 91 [1]). Автор подобрал русскоязычные примеры для наглядной демонстрации основ грамматики связей. Далее будет обсуждаться грамматика для русского языка, принадлежащая классу грамматик связей и разработанная автором в том смысле, что автор придумал словарь требований на связи и коннекторы для русского языка, а не саму концепцию грамматики связей.

3 Грамматика связей и грамматика зависимостей

Прежде чем перейти к обсуждению грамматики связей для русского языка необходимо остановится на отличиях грамматики связей и грамматики зависимостей (по определению Мельчука [3, 4]). Грамматика связей не является ни грамматикой составляющих, ни грамматикой зависимостей. Отличие от грамматики составляющих очевидно, однако с грамматикой зависимостей есть некоторое сходство. Грамматика связей - контекстно-свободная грамматика. В отличие от грамматики зависимостей в грамматике связей связи между словами не имеют направления, могут образовывать циклы, а сами связи разбиты на иерархические классы, впрочем как и слова. В грамматике зависимостей присутсвует корневое слово, которого нет в грамматике связей. Если бы не циклы, то можно было бы сказать, что грамматика зависимостей является вырожденным случаем или упрощением грамматики связей, в случае когда разрешен только один тип коннекторов. Кроме этого автору не известен алгоритм разбора для грамматики зависимостей. В принципе можно попробовать применять алгоритм разбора грамматики связей, в случае коннектора одного типа - но для эффективной работы алгоритма потребуется словарь сравнимый с числом слов в языке - то есть концепция грамматики зависимостей с точки зрения автора неэффективна. Под эффективностью понимается наличие алгоритма, который может разбирать предложение за полиномиальное время, причем число правил в грамматике должно быть значительно меньше числа слов в языке. Неэффективность формализма влечет либо невозможность разбора предложений длинной более 15-20 слов, либо требование составить огромный словарь правил или словарь сочетаемости слов. Базовый алгоритм грамматики связей способен разбирать предложения длиной до 50 слов, не разбивая предложения на сегменты и используя свыше десяти тысяч правил для каждого слова. Эффективность разбора в сочетании с небольшим числом правил, требуемых для описания языка, и есть главное преимущество грамматики связей перед другими лингвистическими формализмами. В общем случае скорость работы алгоритма ограничена константой $O(n^3)$ от числа слов в предложении.

В работе [2] более подробно обсуждаются отличия грамматики связей от других формализмов.

4 Грамматика связей для русского языка

Автор разработал грамматику связей для русского языка, состоящую из нескольких сотен базовых правил и более двух десятков базовых типов коннекторов. Автору пришлось отказаться от классического явного описания грамматических связей для избежания конфликтов с определениями других лингвистических формализмов. Вместо этого используются неявные определения. Это значит, для того чтобы узнать, что означает тот или иной коннектор, нужно посмотреть на примеры его использования. Свобода выбора отношений позволяет повысить коэффициент правильных связей между словами. Рассмотрим типичные списки предложений, используемых для описания коннекторов.

коннектор	предложения
S:	**Полиция ведет** расследование. **Молодые хотят** жить.
	Многие определенно **хотят**. **Меч** всегда **найдется**
EI:	**Немедленно исправить** ошибку.
	Поэтому должен умереть. **Лучше слушай** дальше
MV:	Полиция **ведет расследование**. Немедленно **исправить ошибку**.
	Слушайте нашу **программу**.
A:	Слушайте **нашу программу**. **Алые цветы** смерти. **Иного выхода** нет.
AI:	Один **станет синим**. Мне **было хорошо**. Здесь **все такое**.

Связанные описываемым коннектором слова выделены жирным шрифтом. Полная документация по базовым коннекторам доступна в интернет [2].

Каждое слово дополняется суффиксом, согласно его морфологическим признакам. Это дает возможность при разборе определять часть речи, число, род, падеж. К примеру *buma.ndfsi* означает существительное (n), неодушевленное (d),женский род (f),единственное число (s), именительный падеж (i). А *buma.afss* - прилагательное (a), женский род (f), единственное число (s), краткая форма (s).

Требования по связям определяются не для каждого слова, а для классов слов. В качестве конструктора классов слов используется морфологический словарь www.aot.ru. Классы задаются одним или несколькими признаками и образуют иерархию из более тысячи элементов. Для каждого из них задаются свои требования по связям. Около сотни высокочастотных слов вынесены в отдельный базовый словарь с отдельными правилами требований по связям. Таким образом каждое такое слово образует свой класс.

Каждая формула требований по связям переводится анализатором в специальную дизъюнктивную форму. Формула из 10 элементов после преобразования в дизъюнктивную форму потенциально может образовать формулу из 1000 слагаемых. Для некоторых модальных глаголов формулы требований по связям могут содержать более ста тысяч слагаемых. Поэтому скорость анализатора грамматики связей напрямую зависит от числа доступной памяти. Если все слова русского языка со всеми их требованиями влезали бы в оперативную память, то скорость алгоритма на современном P4-2Ггц составила бы около 100 предложений в секунду. Однако из-за большой прожорливости русских словарей, приходится генерировать словарь из ограниченного числа слов при разборе каждого нового предложения, что неприменно сказывается на скорости - в среднем около 1-го предложения в секунду при потреблении 200 мбайт ОЗУ.

В качестве главного критерия максимизации при разработке грамматики и анализатора было выбрано следующее: правильно разбирать максимально большое число предложений из тестового корпуса предложений. Корпус текстов был предварительно получен из большого корпуса художественных текстов, путем отсева предложений числом слов меньше 5 и больше 25. Под *правильностью* разбора понималось нахождение правильных соответствий между словами в предложении. Предложение не должно разбираться более 10 секунд, а потребление памяти не должно превышать 200 мбайт. Если анализатор превышает ограничение - то включается *режим паники*, в котором резко сужается грамматика - и анализатор быстро находит первый

[2] http://sz.ru/parser/doc

случайный разбор, который часто оказывается правильным.

При сравнительном тестировании с другими анализаторами [3] было обнаружено сравнимое качество разбора. К примеру на одном из подкорпусов [4] при разборе 200 предложений длинной 7 слов поверхностно-семантический анализатор aot.ru правильно разобрал 142-152 предложения (иногда непонятно, что считать правильным разбором), а анализатор грамматики связей около 103-110, и это при том, что в текущей версии анализатора грамматики связей отсутствуют семантические словари и словари валентностей глаголов, которые потенциально могут резко улучшить коэффициент разбора.

Далее мы рассмотрим перловый модуль грамматики связей для русского языка, с помощью которого можно решать практические задачи семантического анализа русскоязычного текста, и в том числе создавать семантические словари в полуавтоматизированном режиме.

5 Lingua::RU::LinkParser

Перловый модуль грамматики связей представляет собой объектно-ориентированный интерфейс, который можно использовать из программ на Perl. После подключения модуля можно работать с объектами соответствующим предложениям, связкам, связям и словам.

Давайте рассмотрим следующую программу:

```
use Lingua::RU::LinkParser
my $parser = new Lingua::RU::LinkParser; # Создаем анализатор
my $text = ‘‘Бита была бита’’
my $sentence = $parser->create_sentence($text); # разбираем предложение
my $linkage = $sentence->linkage(1); # использовать первую связку
print $parser->get_diagram($linkage); # выводим диаграмму на экран
```

Эта программа выводит на экран диаграмму связки:

```
    +---------Wd--------+
    |          +---Sf3--+--AIfi--+
    |          |        |        |
LEFT-WALL бита.ndfsi была.vp бита.afss
```

Первая “бита” успешно распознана как существительное, а вторая - как краткое прилагательное.

Диаграмма помогает нам понять грамматику связей, но для практических задач нам потребуется работать с самим связями.

Продолжая писать начатую выше программу, мы извлечём из объекта $linkage (связка) массив объектов слов $word:

```
my @words = $linkage->words;

foreach my $word (@words) {
   print ‘‘\’’’’, $word->text, ‘‘\’’\n’’;
   foreach my $link ($word->links) {
      print ‘‘ тип связи ‘’’, $link->linklabel,
         ‘‘’ со словом ‘’’, $link->linkword, ‘‘’\n’’;
   }
}
```

Выдержка из вывода программы:

[3] http://www.aot.ru

[4] Корпуса предложений для тестирования http://sz.ru/parser/html4

“бита.ndfsi”
тип связи ‘Sf3’ со словом ‘2: была.vp’
“была.vp”
тип связи ‘Sf3’ со словом ‘1: бита.ndfsi’
тип связи ‘Wd’ со словом ‘0: LEFT-WALL’
тип связи ‘AIfi’ со словом ‘3: бита.afss’
“бита.afss”
тип связи ‘AIfi’ со словом ‘2: была.vp’

Знание части речи и связи для каждого слова позволяет нам использовать грамматические конструкции в программах анализа текста. Давайте рассмотрим одну из них.

6 Задача составления словарей

Эффективность алгоритма разбора грамматики связей позволяет решать задачу составления словарей, через использование *“широких”* грамматик. Предположим, мы хотим составить словарь глаголов с управлениями. Мы можем анализировать большой корпус русского языка и собирать статистику связей глаголов, сколько их и в каких падежах стоят зависимые слова. Например, мы можем составить список глаголов, которые требуют (или могут использовать) творительный падеж. Вот как примерно может выглядеть Perl код, фильтрующий из большого корпуса все связи глаголов к словам, стоящих в творительном падеже:

```
use Lingua::RU::LinkParser;

while (<>)
        $input = $_;
        chomp($input);
        my $parser = new Lingua::RU::LinkParser; # Создаем анализатор
my $sentence = $parser->create_sentence($input); # разбираем предложение
my $linkage = $sentence->linkage(1); # использовать первую связку
        my @words = $linkage->words;
        # для каждого слова в связке
foreach my $word (@words) {
        # для каждой связи в слове
                foreach my $link ($word->links) {
                        # связь должна быть от глагола и в творительном падеже
                        if ($link->linklabel =~ /at/ and $word->text =~ /\.v/) {
                                print ‘‘\’’’’, $word->text,  # печатаем глагол
                                ‘‘ ‘‘, $link->linkword, ‘‘\’’, # зависимое слово
                                $input\n’’;   # исходное предложение
                        }
                }
        }
}
```

Выдержка из вывода данной программы при анализе некоторого русскоязычного корпуса текстов:

"въехал.vsndpms носом.ndmst" Корабль **въехал носом** в одну из воронок и стал кувыркаться .
"цеплялся.vnndpms брюхом.ndnst" Ракетоплан **цеплялся брюхом** за деревья .
"помогли.vsndpp им.m3nst" Время и отчасти землетрясение потрудились над ним , но преуспели мало , ибо люди **им** не **помогли** .
"причиталась.vnndpfs им.m3nst" Вместе **им причиталась** целая свинья .

Два последних разбора, к сожалению неверны. Как оказалось в некоторых случаях анализатору не хватает информации, и если зависимое слово омонимично, имеет одинаковую форму в разных падежах (“ей”, “им”, “всем”), а анализатор текущей версии разрешает связи любого падежа для всех глаголов, то в варианты разбора попадают неверные (с точки зрения носителя языка) разборы. Однако мы можем, используя информацию о числе омонимов, отсечь и такие варианты. Мы возможно потеряем часть правильных вариантов, однако вывод почти не будет содержать неправильные варианты. При составлении словаря для нас прежде всего важно качество, а размер словаря можно улучшить за счет большего анализируемого корпуса. Улучшенная часть кода будет выглядеть так:

```
        foreach my $link ($word->links) {
                (my $main, my $suffix) = split(/\./, $link->linkword);
                my @omonims = grep { /$main/ } gendict($input);
                # связь должна быть в творительном падеже, от глагола
                # и у зависимого слова не должно быть омонимов
                if ($link->linklabel =~ /at/
and $word->text =~ /\.v/ and $#omonims<1 ) {
                                print ``\''''', $word->text,  # печатаем глагол
                                `` ``, $link->linkword, ``\'', # зависимое слово
                                $input\n'';   # исходное предложение
                }
        }
```

Небольшая часть вывода улучшенной версии программы:

"въехал.vsndpms носом.ndmst" Корабль **въехал носом** в одну из воронок и стал кувыркаться .

"цеплялся.vnndpms брюхом.ndnst" Ракетоплан **цеплялся брюхом** за деревья .

"пришла.vsndpfs весной.ndfst" Этой **весной пришла** пора выдавать замуж его младшую дочь .

"почитали.vnpdpp баранами.nlmpt" Люди из усадьбы **почитали** закон **баранами** и благовониями , люди из деревни приносили ему в жертву черепашьи лапки и просяные зерна .

"расписывали.vnpdpp словами.ndnpt" Крестьяне **расписывали** горшки теми же **словами** , которые употреблялись на скалах для докладов древним богам .

"стал.vp молитвой.ndfst" Отчет о процветании **стал молитвой** о куске хлеба .

"осыпали.vnpdpp стрелами.ndfpt" Они **осыпали** жителей **стрелами** , а потом перескочили на стены .

"поменялись.vsndpp мечами.ndmpt" Они **поменялись мечами** , и молодец присоединился к нам .

"завертел.vspdpms мечом.ndmst" И **завертел мечом** , не допуская дальнейших разговоров .

"считали.vnpdpp колдуном.nlmst" Я не возражаю , чтобы вас **считали колдуном** .

"помаргивая.vnnddn линзой.ndfst" Каменный шип порвал куртку , и ствол лазера высунулся из прорехи , предательски **помаргивая линзой** .

Получаемый словарь глаголов, возможно, также содержит ошибки, однако они намного более редки и/или неочевидны, но и их можно выявлять и фильтровать теми же методами, работая с объектами слов и связей. Например, мы можем ограничить расстояние между двумя словами, имеющими связь, ограничить классы глаголов или запретить инверсные связи (когда зависимое слово стоит перед глаголом).

Используя анализатор с более широкой грамматикой, с большим числом разрешенных грамматических явлений, мы можем в автоматизированном режиме составлять различные семантические словари сочетаемости, которые могут использоваться в различных прикладных задачах, и в том числе при создании новой, более узкой грамматики и анализатора связей, который будет более эффективен в практических применениях.

"Использование LinkGrammar для русского языка представляется невозможным" - фраза из обзора защищенной диссертации 2004 года. Автор надеется, что таких утверждений в научных работах после публикации данной статьи не будет. В интернет по адресу http://sz.ru/parser/ демонстрируется работающая версия грамматики связей, также известная как Link Grammar for Russian.

Автор выражает признательность своему научному руководителю Владимиру Васильевичу Рыкову за неоценимую помощь, оказанную им при подготовке статьи.

Список литературы

[1] Sleator D. Temperley D. Parsing English with a Link Grammar. // Carnegie Mellon University Computer Science technical report CMU-CS-91-196, October 1991.

[2] Sleator D. Temperley D. Parsing English with a Link Grammar. // Third International Workshop on Parsing Technologies.

[3] Mel'cuk I.A. Studies in dependency syntax. // Karoma Publishers, Ann Arbor, 1979.

[4] Mel'cuk I.A. Dendency Syntax: Theory and Practice. // State University of New York Press 1988